EDB

7 Critical Success Factors for Moving to Open Source Databases (like Postgres)

Contents

INTRODUCTION

Businesses across industries are seeing the increased adoption of new technologies designed to empower digital transformation and future-proofing efforts. As this trend becomes more popular, the pressure to join in is only growing.

For many organizations, the largest obstacle standing in the way of true digital transformation are the legacy systems on which they have long relied and invested in. While these database solutions might have seemed ideal in the past, their comparative inflexibility, costliness and limited integration capacity makes them poor substitutes for truly innovative database strategies. In a world where it feels more and more like digital transformation or market irrelevance are the only two options, the cost of sticking with these systems is far more than their exorbitant licenses.

So, what is the alternative? Those enterprises who have most effectively embarked on their digital transformation journey have almost universally invested in database technologies that prioritize agility, scalability, freedom and full control of their data. For many, these critical capabilities have manifested in open source solutions, like Postgres database.

Open source databases, such as Postgres, offer the greatest flexibility in how enterprises unlock the power of data, while continuing to offer the robust functionality required for enterprise-ready databases. With the backing of dedicated and innovative communities, these database solutions have proven to be ideal launching pads for digital transformation, suited to the needs of developers, architects and IT personnel alike. In fact, **80% of IT leaders** today expect to increase their use of enterprise open source software for emerging technologies.

As the conversation around digital transformation evolves, the benefits of open source and Postgres are an ever-present topic of interest.

In this guide, we'll explore why open source is so valued among modern businesses and outline the key factors that can ensure your success as you plan your move to an open source database like Postgres.

1. Identify your business case

Identify your business case

Before you start looking into open source databases to decide which is right for you, you must identify why you are moving your data and applications, how you plan to do so, and what you're expecting to accomplish. The answers to these three questions will help you build the foundation of your strategy and ensure you have metrics in place to gauge the success of your efforts.

What are your drivers?

The first area to explore is your own motivations for undertaking a move to open source. While the broad answer might be "digital transformation," you want to iterate on this notion a bit more more, in order to help you design the ideal plan for your future.

Think about why you are looking at open source specifically

As previously mentioned, there are a variety of reasons that so many businesses have decided to invest in open source options like Postgres over legacy database options. If your organization is truly invested in acceleration and transformation, open source is a powerful technology to bet on, both because of its flexibility and the thriving community that ensures constant innovation. In fact, open source mandates have become common across many enterprises, including the European Commission, which recently adopted new rules for open source software distribution, motivated in part by the ability to publish source code easier and more efficiently.

There's also Postgres' ability to integrate with a wide variety of tools, ensuring that your workflow continuity. This is one of the key differences between proprietary legacy software and open source technology. While the former places restrictions on how you can work—slowing your business and inhibiting your growth—the latter places an array of tools and opportunities at the fingertips of every person in your business, empowering you to innovate, to go faster and further. With open source, technology is being distributed in a meaningful way that's well-suited to any organization building dynamic applications. This philosophy is the beating heart of modernization and digital transformation.

Postgres is the most loved, most used, and most wanted database in the world.

What benefits could you see from Postgres?

Powering innovation through community, tapping a distributed talent network tosupercharge innovation – Postgres does all of this at a dramatically lower cost than legacy, proprietary systems. Postgres is the **most transformative open source technology since Linux**, Why? Because data, and in turn databases, are strategic to business, andPostgres supports more enterprise applications than any other open source database. It is a performant, scalable, reliable, and secure database solution that enables you to focus on building business logic without having to waste time on things like slow, unreliable database provisioning, production database stability issues, or maintenance downtime. Postgres is taking off because enterprises globally are realizing that they can move all of their database applications to an open source platform.

And those that take advantage of Postgres don't regret it. Combined, Postgres is the most loved, most used, and most wanted database in the world, according to **StackOverflow** surveys of developers, positioning it for exponential growth in 2022 and beyond. The chart below shows that it's been trending this way for the last few years, and for the first time, Postgres now leads in all three categories.

What benefits could you see from the cloud?

Many organizations will have deep sunk investments in Oracle, Microsoft or IBM relational database management systems (RDBMS) and would like to relax that dependency in order to avoid vendor lock-in, reduce costs and gain access to databases designed for the modern world of private and public cloud, Big Data analytics and flexible tooling ecosystems.

One obvious way to achieve this is widespread adoption of cloud technology to access the latest software, gain elastic compute capacity, support mobile devices, shrink hardware cost bases, and reduce IT administration overheads. Cloud adoption enables IT leaders to re-platform applications, which then opens the door to removing clunky requirements such as incumbent database platform providers or net new development. Cloud is also widely seen as an offramp for those aforementioned older applications by refactoring to embrace the new in the form of microservices and to shave off risks such as software being no longer supported. This is far from easy, but many take a middle road which sees them deconstruct applications, placing more services in enterprise environments in an extended iterative process.

Which database strategy will you choose?

Lift and Shift

With a lift and shift strategy, you're taking an on-premises architecture, and moving it wholesale, unchanged into the cloud, without optimizing it for the new cloud database—no microservices, no re-engineering. Those who choose the lift and shift approach will frequently mention how fast, easy, and cost-efficient it is, all of which is true. However, this strategy is only suitable if there's pre-existing compatibility between your on-premises environment and your new cloud environment, i.e. if you're moving from Postgres to Postgres in the cloud. In other cases—such as moving from Oracle databases to Postgres in the cloud—you'll need additional tools to ensure compatibility, like **EDB Advanced Postgres Server** (EPAS).

Replatforming and Restructuring (Transformation)

In this scenario, you're focused on making changes to existing applications to make them more compatible with either cloud or open source. Replatforming is often a question of making an architecture more scalable or adding new capabilities to existing applications. There are additional components that come into play here. Consider an organization that wants to go to Kubernetes, because they're interested in the extra capacity Kubernetes can provide their application servers. In order to actually achieve this, they need to restructure, often using microservices, to make that transformation.

But you can also use replatforming to "transform in place." Many businesses looking to move to a new database will deploy a lift and shift approach and begin transformation once they've reached their destination, rather than doing so prior.

New Applications (Start from Scratch)

This final approach is most often used by businesses that have applications that are not providing value or would simply be too work-intensive to refactor or restructure because of their age. While starting from scratch on new applications might seem like the most complex of these approaches, it's often less of a drain on resources than trying to salvage or rebuild applications that simply aren't compatible with your new environment.

It's important to note that restructuring is for when you have something of value that's worth the effort. In many cases, building new applications is more worthwhile.

What are your desired outcomes?

Once you've established your drivers and your strategy for moving to open source, it's worth asking yourself how you plan to gauge the success of your project. Of course there's the basic goal of successfully moving all your assets and applications into your chosen environment, but businesses also want to ensure that their Postgres environment consistently runs on par with the systems with which they were previously accustomed.

What long term success looks like to you ultimately ties back to your drivers:

- Are you striving for increased adoption of your applications by customers?
- Are you looking to expand your customer base by a certain percentage or maybe prevent the loss of existing customers to competitors?
- Are you looking to maintain, restructure, or totally rebuild your central applications?
- What sort of commands from your applications do you need your database to respond to?
- What non-functional requirements (more on this later) are you prioritizing? Scalability? Security? High-availability?

Even if the answer is as simple as growing your revenue or "all of the above," knowing how you'll measure your success is key to figuring out your priorities for your new database, and, asa result, how you'll configure it.

2. Understand your database requirements

Understand your database requirements

Once you've established why you're moving to an open source database environment, you need to figure out what that database will look like, how it will function, and exactly what you need from that database to enable transformation. Considering how many businesses have opted to use Postgres, we'll use that as our central example.

Deployment technologies: automation and control

When it comes to a Postgres database, one of the most important questions to ask is: how much of the management responsibility do you want to shoulder? The more responsibility you take on, the more control you'll have. However, this also means a great deal of work for IT and operations teams.

Let's look at the three main deployment options, and how each affects your database requirements.

Virtual Machines and IaaS

As it stands, using virtual machines requires the most internal management of the three options, demanding both an infrastructure architect and an infrastructure DBA (database administrator). While for some, this sounds daunting, the VM approach is well-regarded by those who want the most control possible over their architecture, as this leaves infrastructure essentially entirely in your hands.

Another benefit of this approach is that your architecture does not change from a traditional on-prem architecture, which makes it easier to move applications and make use of critical solutions without much of a learning curve.

Kubernetes

While **Kubernetes** still requires your business to have an infrastructure architect, the need for a DBA is greatly reduced, because your Kubernetes system is managed by an operator, freeing you from much of the responsibility that comes with VMs. This does limit architectural flexibility slightly, which can be frustrating for those who abandoned a legacy database specifically because of the restrictions on their business. Additionally, in order to maintain your architecture in its on-prem state, you'll need to invest in a microservice transformation, which can be a slight headache.

Database-as-a-Service (DBaaS)

Finally, you have the DBaaS approach. In this case, you need neither an infrastructure architect nor an infrastructure DBA, as all infrastructure is the responsibility of the service provider, such as EDB. While this sacrifices some flexibility, the DBaaS strategy boasts much more elasticity and on-demand capabilities than either other option. Additionally, any changes to your architecture from its on-prem form are minimal, and you aren't tasked with ensuring architectural continuity.

For businesses who are less concerned about being able to tailor every aspect of their cloud architecture, the DBaaS approach is incredibly popular, and also makes it much easier to channel resources that might have been spent on infrastructure management to innovation and consolidation. It's renowned for its flexibility, ease of travel deployment, low-admin footprint and cost-effectiveness.

Functional requirements

When considering DBaaS solutions, functional requirements are an essential component. At their simplest, a functional requirement relates to what commands your applications will be sending to the database, and what services the database will have to provide in return. Common examples of this include creating new employees, compiling payroll data, or changing salaries. When you hear a provider talk about categories or distinctions such as:

- **Relational vs. Non-Relational**
- **SQL vs. NonSQL**
- **Schema vs. Non-Schema**
- **Structured vs. Unstructured**

They're referring to types of databases and how they will communicate with your applications. Digging into what your applications will need your database to be able to perform will ultimately govern which of these you'll invest in.

Non-functional requirements

In contrast to functional requirements, non-functional requirements refer to how fast a database can perform those tasks discussed above, whether it will be able to perform multiple functions at once, and whether it will be able to do so consistently.

Under this category, you find considerations such as scalability, high availability, performance, and even security. All of these relate, not to whether your applications and database can send and receive commands, but how effectively they can do so.

3. Determine your adoption path

Determine your adoption path

At this point, it's essential to dive a little deeper into the different strategies you're considering for your move to Postgres. Each option at your disposal is frequently motivated by a common set of drivers, and demands a different set of ways to analyze its efficacy. In short, this is the culmination of the three questions we posed in the first section of this guide.

A good way to understand what choices you have at your disposal is through what are known as the **5 R's of Rationalization.**

The 5 R's of rationalization

EDB is a database company to the core; we know PostgreSQL, and we have the talent and the technology to bring something new to the table in database-as-a-service (DBaaS).

Rehost

Rehosting is most easily mapped onto what we earlier referred to as "lift and shift." It involves moving your assets to a new database provider, while changing the overall architecture of those assets very little.

Why rehost?

- Cutting spending
- Freeing up space
- Taking advantage of new features or technologies

What to consider:

- Network traffic
- Asset and application compatibility
- Disruptions to business continuity
- Onboarding challenges

Refactor

When businesses adopt platform-as-a-service (PaaS) solutions, they often choose to refactor the code of their applications to fit a PaaS-based model.

Why refactor?

- Lower update times
- More portable code
- Increased database and/or cloud efficiency

What to consider:

- How large are your applications and assets?
- What is your network bandwidth?
- How much CPU does your database have?
- Do you have major business dependencies?

Rearchitect

This approach is most commonly used for businesses looking to deploy Postgres in the cloud. Due to their architecture, some older applications aren't compatible with cloud providers. In other cases, applications might be cloud-compatible but not cloud-native. This can create operational inefficiencies.

In both these scenarios, rearchitecting refers to updating the architecture of the application to ensure its compatibility or cloud-native nature.

Why refactor?

- Adopt new cloud capabilities
- Ensure agility and flexibility of applications
- Enable a storage environment containing multiple stacks or databases

What to consider:

- How large are your applications and assets?
- What is your network bandwidth?
- How much CPU does your cloud database have?
- Do you have major business dependencies?
- What are your development languages?
- What strain might this put on DevOps?
- Do you have major business dependencies?

Rebuild

In cases where carrying an application forward will simply be far more expensive than the value the application provides, businesses will opt to rebuild those applications with a new code base that aligns with their new DBMS.

Why rebuild?

- Reduce operational costs
- Prioritize innovation and digitization
- Build applications faster

What to consider:

- How large are your applications and assets?
- What is your network bandwidth?
- How much CPU does your database have?
- Do you have major business dependencies?
- What are your development languages?
- How much will it cost to rebuild?
- How will end users react?
- How severely are applications limiting your business processes?

Replace

For businesses looking to implement the best possible applications, it sometimes makes sense to invest in a pre-built solution and ditch the applications that you, yourself have created. In scenarios like this, you can either do so immediately, or tag an application for future replacement, and remove it from your transformation project.

Why rebuild?

- Standardizing around industry best practices.
- Accelerating adoption of business-process-driven approaches.
- Reallocating development investments into applications that create competitive.
- differentiation or advantages.

What to consider:

- How will this reduce or increase operating costs?
- What's the cost-benefit of my current application architecture versus that of a SaaS application?
- How many assets do I need to retire?
- How will this affect my automated processes and workflows?

4. Get your team ready

Get your team ready

When moving to an open source database, there will be a range of changes and shifts you'll experience. Because the process is a complex one that requires meticulous attention, it's vital to prepare your team for what's to come.

Depending on your team, this can take many different forms. However, open communication is essential. This will help you establish who will need training and in what capacity. Everyone should know their responsibilities going into a transformation project, otherwise even the simplest elements can go awry.

Who to consider

Architects

It's important to talk with your architects about how the architecture of your assets will change in your new environment. They'll play a significant role in testing that new architecture once your transformation is completed, so they need to be extremely familiar with what they'll be navigating or reshaping. Otherwise, essential assets could get lost in the shuffle, and business continuity can suffer. While many open source databases like Postgres are designed to allow you to deploy your architecture more or less unchanged from its previous form, others require significant tweaks that architects will have to be aware of.

Developers

When it comes to moving to an open source database, developers don't play that big of a role in the move itself. Their primary responsibility will be in testing applications once the move is complete, which itself is of vital importance. As we'll discuss later, ensuring that your applications are working properly in the wake of a transformation project can affect a wide range of business elements, including security and compliance, which can never be taken for granted.

Third party assistance

When executing a move to open source databases, it often helps to invest in outside expertise. At EDB we've worked with numerous businesses deploying Postgres to help them plan and monitor their transformation projects, ensuring greater efficiency and management, community support, and enhanced tooling, portability, and high availability. Investing in a database expert gives you constant support, so that, if you do run into any issues, you don't have to figure out how to solve them yourself. This frees up IT and operations to work closely with your internal team, rather than devoting resources and time to correcting an error that an expert in your chosen database is better qualified to take on.

Downtime and Business Continuity

One of the biggest factors in preparing your team for a migration or a move is how much time you have at your disposal to actually execute the migration.

Imagine you have 5 TB of data that you want to move between Oracle and Postgres. This process can take anywhere from one to two weeks. So, while you may have all the resources and capabilities to perform this migration, you may not have the downtime window to handle applications being offline or unavailable for that long.

One way to deal with this is an approach called "Change Data Capture." Here, you make a copy of all your data to migrate, and then apply all the changes that have been made to the initial data over the two weeks that migration took to the data now in your new environment. Change Data Capture is one of the best ways to navigate downtime issues and is available through EDB's own **Replication Server**.

Factoring downtime into your strategy for moving to Postgres is crucial because, very often, mission-critical databases simply cannot be down long enough for you to just migrate all of your assets. This is why we've seen so much success with Change Data Capture-backed migrations. They allow for the migration to go through without disruptions that might harm business continuity. It's this approach that makes possible what we like to call **Always-On Postgres**.

5. Ensure mission-critical capabilities

Ensure mission-critical capabilities

At this point, you're in a position to execute your move to Postgres or your chosen open source database. However, in order to ensure that everything gets where it needs to go, and functions properly once there, you must move the components of your business in specific, strategic waves.

Moving to open source in 3 easy steps

a. DB Schema, Codes, and Data

This step should include moving schema, and migrating DB functionality and data as a snapshot and/or CDC. This will form the basis of your new database infrastructure, and enable all of your interfaces and applications. This phase of your move is also vital to understanding your need for CDC and what your cutover timelines will look like. If there are gaps you need to fill, you'll be able to recognize them here.

b. Interfaces and Applications

During this phase move APIs (JDBC, ODBC, OCI, .NET, …), convert embedded application SQL, and move applications. From this, you'll be able to run an application that meets functional requirements, and integrate the app and database. It will be important to test everything moved during this stage to ensure that the applications are functioning as intended.

c. Reports and Management Tools

Finally, be prepared to move reports, DBA utilities, and script. This will help ensure DBA tools (data loading and other management) are functional and that reports are generated through SQLPlus or other means, e.g., Tableau, Qlik.

Each step of this process is very important; the order in which you carry it out ensures that your architecture is organized, secure, and working. It's critical to pay extra close attention during the first phase of moving, because that will determine the success of everything to come. If your schema, codes, and data aren't in place, your interfaces and applications won't work, which means both your teams and your customers will be unable to make use of the tools they need or the services they expect.

6. Monitor and thoroughly test your apps

Monitor and thoroughly test your apps

Once you've completed your move or transformation into open source, it's essential to test everything you've moved across the board. This will ensure that your most essential applications are working and help guarantee that your digital transformation journey is on the right track.

Functionality benefits

The most obvious benefit of testing your applications—after your transformation project and regularly going forward—is that you'll be able to prevent downtime and address outages more effectively. Whether these are applications you use in your own workflows or ones that are customer facing, you want to guarantee business continuity and maintain customer satisfaction. As a digitally positioned business, you need to prove that you're consistently operational in the digital space.

Security benefits

Applications that help you manage or share vital assets are as much a question of security as anything else. If these applications aren't working properly, you need to quickly understand how and why, otherwise the data that these applications handle might be vulnerable. Breaches in security or issues with handling PII are a nightmare. Testing is one key way to ensure you sleep soundly.

Compliance benefits

As an extension of security, adhering to the various compliance regulations that exist within your industry should be top of mind when building and maintaining any and all applications that might touch sensitive data. While security breaches are already a black mark on a business' reputation, they can also lead to substantial penalties. Violating compliance regulations such as GDPR even accidentally, is a major issue. Regularly testing the applications that might be vulnerable to issues such as that is your best and easiest option to guarantee compliance consistency and avoid disaster.

7. Tap into community innovation

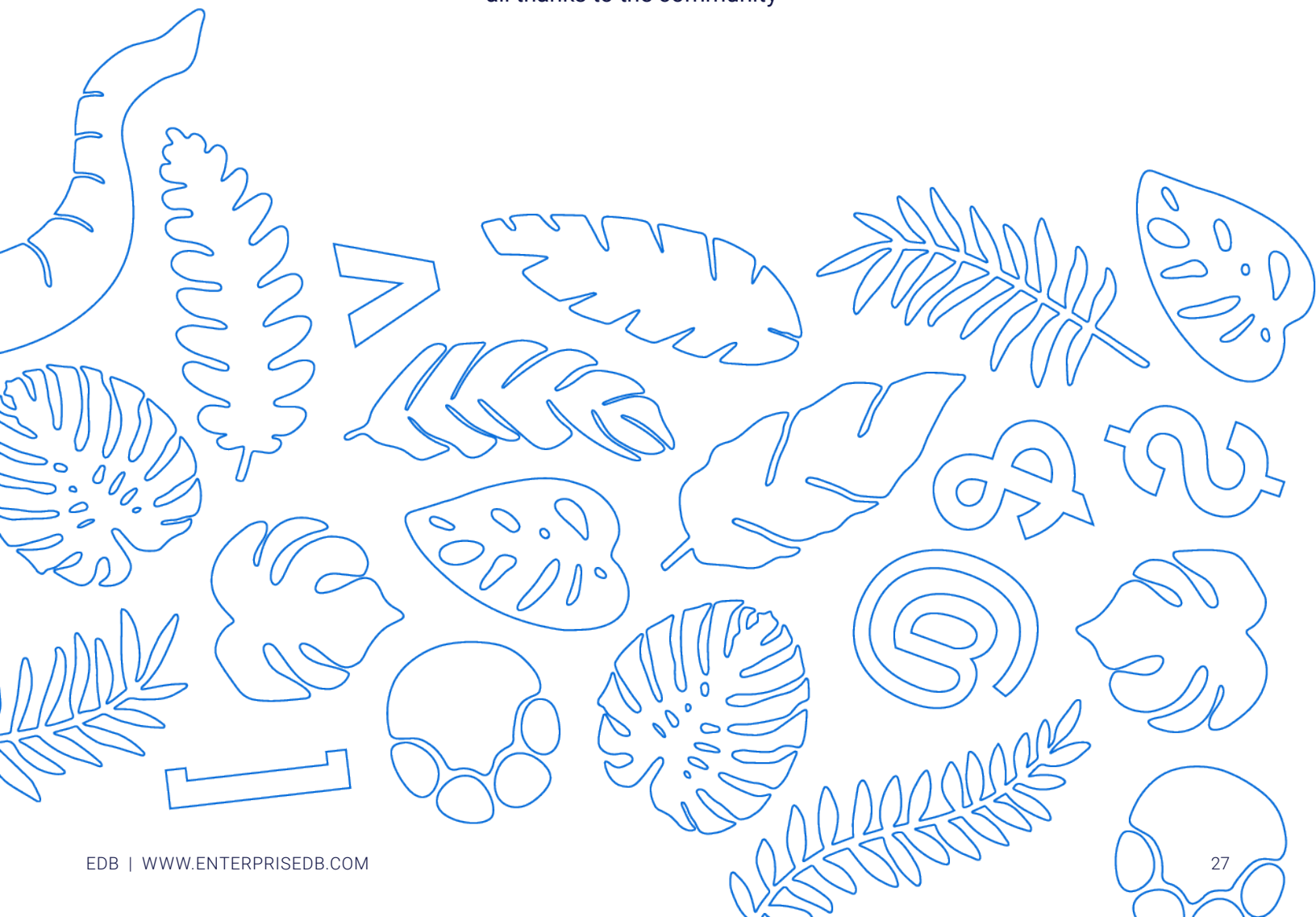
Tap Into Community Innovation

One of the greatest strengths of open source databases is the community that builds it, made up of experts and enthusiasts who are always working to make the solution better. The Postgres community ships significant releases with high quality ona predictable cadence. This means that bugs are identified and fixed swiftly and updates are regularly and easily available.

People come to Postgres for the cost and stay for the innovation.

As you're considering investing in open source, we encourage you to explore all the resources at your disposal. Adjusting to a new DBMS can sometimes feel tricky, but with Postgres you'll find businesses whose needs and drivers align closely with yours, as well as information on exactly how to make the most of your enterprise's new home.

Yes, organizations come to Postgres for all sorts of reasons: cost, flexibility, integration. But, it's the innovation they stay for. And that's all thanks to the community



What's next?

What's next?

These seven critical success factors will ensure you're equipped to move your business technology forward to the future. With clear objectives and the right methods to deploy them; an understanding of your options and which work best for your enterprise; team preparedness and application environment due diligence; and a direct line to the builders and innovators bringing Postgres to the world—you're on the fast track to agility and scalability.

Businesses in pursuit of digital transformation and true innovation have long found that open source is their best answer and the potential of Postgres is limitless. Offering a massive leap forward in flexibility, Postgres has very significant knock-on effects that drive a culture of innovation, and give every member of your organization the freedom to try things. It's the ideal home for digital transformation.

The best part is, you don't have to do this alone. EDB has helped thousands of organizations harness the power of Postgres and we've seen firsthand what they've accomplished as a result. Your enterprise's vision combined with our decades of expertise, the future can be even more exciting than you imagined.

See how leading companies are succeeding with EDB

About EDB

EDB provides enterprise-class software and services that enable businesses and governments to harness the full power of Postgres, the world's leading open source database. With offices worldwide, EDB serves more than 1,500 customers, including leading financial services, government, media and communications and information technology organizations. As one of the leading contributors to the vibrant and fast-growing Postgres community, EDB is committed to driving technology innovation. With deep database expertise, EDB ensures high availability, reliability, security, 24x7 global support and advanced professional services, both on premises and in the cloud. This empowers enterprises to control risk, manage costs and scale efficiently. For more information, visit **www.enterprisedb.com**.

7 Critical Success Factors for Moving to Open Source Databases (like Postgres)

EnterpriseDB Corporation
34 Crosby Drive
Suite 201
Bedford, MA 01730